

A Word About **E.S.R.** Inc.

E.S.R., Inc. was founded several years ago by three scientists and engineers who wanted to provide toys and educational devices that would have those qualities that lead children and adults to fuller and more successful lives — and yet bring enjoyment to all. Their first attempt, DIGI COMP I, was an instant success. Today, tens of thousands of youngsters and adults in America and Canada have enjoyed DIGI COMP and now have been introduced to the computer field. Over a thousand public and private schools have used DIGI COMP for classroom demonstrations. New textbooks in Science and Mathematics are including a discussion of DIGI COMP. With the new advanced manual now available for DIGI COMP a very thorough examination of computer math and fundamentals is now a reality for all.

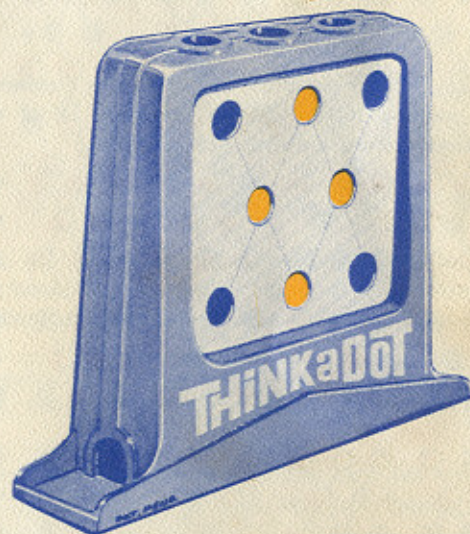
With the success of DIGI COMP, E.S.R., Inc. is continuing in the same direction, providing toys with a purpose. The Amazing DR. NIM is uncanny in its simplicity and skill. Although completely different than DIGI COMP I, it is still a real binary digital computer entirely in plastic which can provide an insight and understanding into the "thinking" ability of man-made machines. THINK-A-DOT is again in the same theme, a binary digital computer entirely in plastic, but uniquely different in its structure. Here is a case of a computer that provides unending fun (even a two-year-old is fascinated), but also challenges the skill of a computer expert.

The scientists at E.S.R. have proven that devices that actually operate on basic engineering principles provide in reality far greater pleasure than pseudo or fake gadgets that insult the intelligence. They have shown that you, the American public, want and in fact prefer the real thing and want to learn while you play. This is the objective of E.S.R. — to provide toys with a purpose.

E.S.R., INC., 34 LABEL STREET, MONTCLAIR, N. J. 07042

THINKaDOT

INSTRUCTIONS



E.S.R.

Start by tilting it on its side . . . then stand it up . . . *Now*, drop a marble in any of the 3 holes on top . . . Try to make all the spots BLUE . . . or YELLOW . . . by dropping the marble in the holes on the TOP over and over . . . That's the FIRST game you can play with THINK-A-DOT. There are lots more . . .

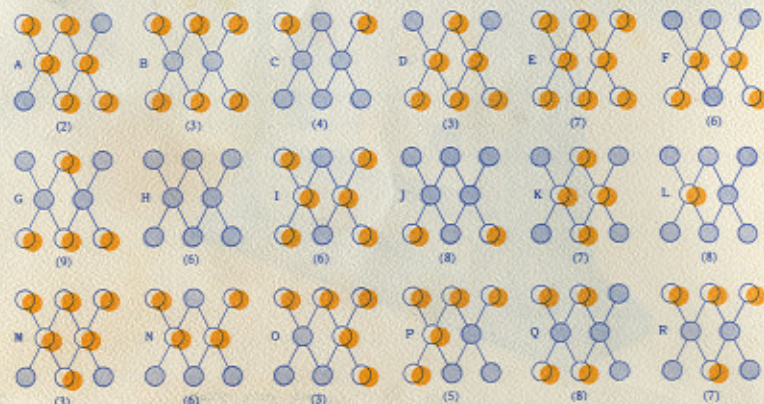
OTHER GAMES YOU CAN PLAY WITH THINK-A-DOT

To Play Alone

Start by tilting THINK-A-DOT to the RIGHT so that the pattern looks like this:



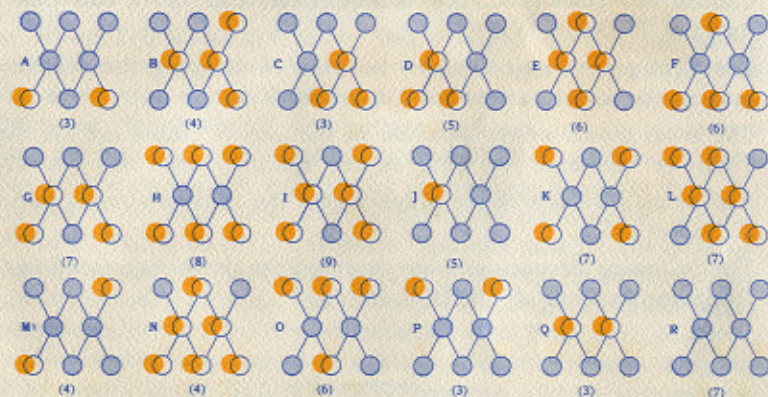
Then try to get any of the patterns below in the number of moves indicated below each pattern. (These are only 18 of a possible 128 patterns. You may work the rest out yourself.)



Next, start by tilting THINK-A-DOT to the LEFT so that the pattern looks like this:



Then try to get any of the patterns below in the number of moves indicated below each pattern. (These are only 18 of a possible 128 patterns. You may work the rest out yourself.)

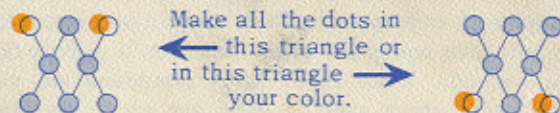


Answers to all of the above moves are given at the end of this instruction book. Try to predict the new dot pattern that will occur with each drop of the marble. This will help your play.

To Play Against A Friend

As a game, two players try to outguess THINK-A-DOT and each other.

Each player first chooses one of the two dot colors as his own. Then they choose a dot pattern as the winning pattern. Two of these might be:



Or choose any of the patterns given so far or any others you want.

Then set THINK-A-DOT to a starting pattern by either tilting it to one side or the other or shaking it to get a random starting pattern. Choose who goes first. The second player chooses a side (left or right) of THINK-A-DOT.

The first player may continue his turn of dropping the marble as long as the marble comes out his chosen side.

The other player starts as soon as the marble comes out his side. Then he continues as long as the marble comes out his side. And so on until one of the players gets the chosen winning pattern all in his color.

The following discussions are intended for those interested in the Computer Aspects of THINK-A-DOT.

Section II

THINK-A-DOT AND COMPUTERS

A computer is a tool for working with information. Usually the information takes the form of numbers and words. The computer can memorize information given to it by people. It can then perform various calculations on this information. Results of these calculations can also be remembered by the computer. A person can find out what is in the computer's memory when needed.

Electronic computers are used today to predict weather and elections, control space shots, and calculate paychecks. These computers can remember large amounts of information (all the records for thousands of employees). They can also operate at very high speeds (calculate and print each employee's paycheck in a fraction of a second). While these large machines are very complicated, the basic ideas used in them are not. Very simple, slow, mechanical computers can be built using the same ideas. These small computers, while not as useful as their larger cousins, are

sometimes more fun. THINK-A-DOT IS ONE OF THESE SMALL COMPUTERS. While intended as a game it also illustrates many of the basic ideas used in larger machines.

By tilting THINK-A-DOT to the left or right you can put one of two different information (or color) patterns into its memory. Note that when you return THINK-A-DOT to its upright position it retains or "remembers" the pattern you set into it when you tilted it. You can then cause THINK-A-DOT to perform calculations on the memorized information by dropping a marble in at the top. The newly calculated color pattern then replaces the original pattern in THINK-A-DOT's memory. You can observe the information (color pattern) in THINK-A-DOT's memory by means of the eight windows provided. The memory of a large computer might comprise millions of individual memory cells. Each cell, however, could remember only one of two things. For this reason, they are called binary cells. THINK-A-DOT contains eight binary memory cells, each of which remembers either of its two colors. Another name for these cells is Flip-Flop.

Much of the ability of a large computer comes from using parts which perform "logical" operations. You can easily demonstrate "logical" operations with THINK-A-DOT. Drop the marble into the middle hole. The bottom right-hand dot will change color only when the top-middle AND the middle-right hand dots are both yellow. This is called a "logical" operation. See if you can discover others in THINK-A-DOT.

Section III

THINK-A-DOT OPERATION

For those interested in learning how to program THINK-A-DOT the following discussion may be helpful.

1. Each time a marble hits a flip-flop it changes its color (State). Hence, these are complementary flip-flops.
2. Since the middle and lower flip-flops change color no matter which direction the marble comes from, we can say that they are connected by "OR" gates. That is to say, a marble coming in from the right OR the left hand path will change (complement) the flip-flop.
3. The direction the marble goes upon leaving a flip-flop depends upon its previous color state. For example, when the center top flip-flop is YELLOW the marble will take the right hand path and move to complement the right hand flip-flop of the center pair. On the other hand, if the top center flip-flop is the BLUE state the marble will move to the left hand path and complement the left hand flip-flop of the center pair. And so on.

NOTE: Not all flip-flops have this same direction routing, of the marble. You will have to experiment with each flip-flop to discover all of the conditions.

4. Finally, the side on which the marble comes out depends on the color (state) of the flip-flop it complements in the bottom row of three flip-flops. For example, if the center bottom flip-flop is yellow the marble will come out of the right hand side, if it hits that center flip-flop. And so on . . . You will have to experiment to find the complete set of conditions.

Now, if you have written down all of the above conditions, you can draw a FLOW CHART or diagram for your THINK-A-DOT from which you can program or predict each move. Each flip-flop acts as an information storage element, as well as a logical control element. The marble acts just as an electric current flowing between these elements through OR gates, just as it might in a real electronic computer!

Section IV

TEN THINK-A-DOT THEOREMS AND COROLLARIES

The previous description of Section III on the mechanics of operation can now be applied to prove the following theorems and corollaries for THINK-A-DOT. You may enjoy proving them yourself either by trial or by analysis. Remember that THINK-A-DOT is made up of eight binary (two-state) flip-flops that are flipped and interrogated for their prior state each time a marble hits them.

1. If a total of "a", "b", and "c" marbles are dropped in the left (L), middle (M), and right (R) holes in the top of THINK-A-DOT, then the new pattern of dots obtained is independent of the order in which they are dropped.
2. If (8, 0, 0) or (0, 8, 0) or (0, 0, 8) marbles are dropped in the L, M and R holes respectively, the pattern will be unchanged. [(8, 0, 0) means, drop eight marbles in the top left hole, zero in the middle hole and zero in the right hole].
3. If (2, 2, 2) marbles are dropped in the L, M and R holes the pattern will be unchanged. This implies that (4, 4, 4) and (6, 6, 6) drops will also leave the pattern unchanged.
4. If the sum of the blue (or yellow) dots in the top and bottom rows, excluding the middle row, is even (or odd) then dropping any number of marbles in L, M or R will leave this total even (or odd).

5. If the starting pattern has an even (or odd) number of blue (or yellow) dots in the top and bottom rows the target pattern must also have an even (or odd) number of blue (or yellow) dots in these rows for it to be an obtainable pattern. Thus, if the parity* agrees, the target pattern can be obtained from the starting pattern. Otherwise it cannot.
6. To obtain a new dot pattern " P_2 ", from a starting pattern " P_1 ", proceed as follows:
 - a. If the top left hand dot in the new desired pattern " P_2 " does not agree with the starting pattern " P_1 " drop 1 marble in L to make it agree. (Otherwise do not drop any marbles in L.)
 - b. Drop as many marbles as required in M to make the diagonal extending downward from M to the left the same as that in the new desired pattern " P_2 ". (No more than 7 marbles will be required.)
 - c. Drop as many marbles as required in R to make the diagonal extending from R downward to the left the same as P_2 . (No more than 7 marbles will be required.) If P_2 may be obtained from P_1 the lower right hand dot will automatically be the correct color after this is done. Otherwise an attempt has been made to transform an "even" pattern into an "odd" pattern or vice versa, which, of course, is impossible.

The resulting number of marbles, (a, b, c), dropped in L, M and R by this technique will be a "feasible" solution to the problem of transforming P_1 into P_2 but not necessarily an "optimum" solution. It will also, as a matter of fact, be a "primitive" solution with these terms defined as follows:

*Parity — merely means that the sum of the dots of one color (often called "BITS") is always either even or odd.

A "feasible" solution, (a, b, c), is a triplet of numbers such that if these numbers of marbles are dropped respectively in L, M and R in any order an initial pattern P_1 will be transformed into a desired pattern P_2 .

A "primitive" solution is a feasible solution with no number greater than 8.

An "optimum" solution is the primitive solution that makes (a + b + c) a minimum.

7. All primitive solutions may be obtained as follows:
 - a. Obtain the first primitive solution by the method described in 6.
 - b. Add (2, 2, 2) to the numbers (a, b, c) obtained and if any number obtained is greater than 8 subtract 8 from it.
 - c. Repeat 7b on each solution obtained until a prior primitive solution is obtained. Exactly 4 primitive solutions will be obtained.
8. The optimum solution may now be selected from the primitive solutions in 7 as the one that minimizes the sum of the number of marbles required.
9. An optimum solution will never require more than a total of 9 marbles.
10. All solutions may be obtained by adding multiples of (8, 0, 0) or 0, 8, 0) or (0, 0, 8) to the primitive solutions.



EXAMPLES OF THE APPLICATION OF THE ABOVE THEOREMS AND COROLLARIES

- Theorem # 1. Tilt THINK-A-DOT to the right. A blue diamond will appear in the center with the 4 corners yellow. Put 3 in L, 2 in M and 1 in R. This should transform the pattern to all blue. Now tilt to right again and put marbles in THINK-A-DOT in the order L, M, R, L, M, L. The same pattern should appear.
- Theorem # 2. Start with any pattern you wish. Then drop 8 marbles in any one hole and see if the same pattern returns.
- Theorem # 3. Drop (2, 2, 2) marbles in L, M, R and see if the same pattern returns.
- Theorem # 4. Verify Theorem 4.
- Theorem # 5. Tilt to right. Drop marbles in any hole and note that the number of blue dots always remains even.
- Theorem # 6. Tilt to right. Try to make all dots blue by the method described in Theorem 6. This should take (1, 0, 7). Now try to obtain all yellow from all blue.
- Theorem # 7. Note that $(1, 0, 7) + (2, 2, 2) = (3, 2, 9) = (3, 2, 1)$. Thus, (3, 2, 1) will also transform the original pattern to all blue. Find all primitive solutions.
- Theorem # 8. Verify that (3, 2, 1) is the optimum solution.
- Theorem # 9. If (1, 7, 5) is a solution what is the optimum solution? Note that the maximum of 9 marbles is required for the optimum solution.

HOW THINK-A-DOT CAN BE USED AS A COMPUTER TO COUNT, ADD, AND SUBTRACT

We will now be able to let you prove to yourself that THINK-A-DOT is really a binary digital computer by explaining how it can count, add and subtract. In order to do this we will first examine number systems so that you may learn how to communicate with your THINK-A-DOT.

This discussion is not intended to be complete. We expect you to work out additional programs and techniques for using THINK-A-DOT.

Number Systems

All digital computers operate in a number system known as the Binary System. The primary reasons for the use of such a system are its simplicity and the fact that it can be so easily mechanized. There are only two digits, "0" and "1" (called bits) in the system. Common examples of this simplicity and use are:

An electrical switch which is either on or off; a particular location on a computer punch card which either has a hole or does not; or a mechanical flip-flop which either shows blue or yellow as in THINK-A-DOT.

In each case, the device has just two stable, mutually exclusive conditions. The digit value 0 is assigned to one of these conditions and 1 to the other.

In ordinary use, binary numbers would be quite cumbersome, since about 3-1/3 binary bits are required to represent a single decimal digit. This represents no problem to the mammoth electronic com-

puters because of their high speeds and large number of electrical components.

Before going into these other number systems in great detail, we should review the inner workings of the *decimal system* (base 10). The central feature of the decimal or arabic number system is the fact that a given digit can have more or less value depending on where it is written. For instance, the symbol 2 may mean 2.0, 200, or .0002 depending upon where it is written with respect to the decimal point. This is so familiar to us as to be overlooked in everyday use of decimal numbers. For example, what we really mean by 2756 is, of course:

$$2000 + 700 + 50 + 6 = 2756$$

$$\text{or } 2 \times 10^3 + 7 \times 10^2 + 5 \times 10^1 + 6 \times 10^0 = 2756 = (2756)_{10}$$

This is the basis for calling the decimal system a "base 10" system.

In the *binary system* (base 2) the number $(1101)_2$ would actually mean:

$$1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$$\text{or } 1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1 = (13)_{10}$$

The subscript at the end of the parenthesis indicates the base of the number system. Thus $()_2$ means the base 2 or the binary system. Or $()_{10}$ means the base 10 or our familiar decimal system.

Similarly, in an *octal number* (base 8) system, where there are 8 characters (the digits 0 thru 7) a number $(6573)_8$ would mean:

$$(6573)_8 = 6 \times 8^3 + 5 \times 8^2 + 7 \times 8^1 + 3 \times 8^0$$

$$\text{or } 6 \times 512 + 5 \times 64 + 7 \times 8 + 3 \times 1 = (3451)_{10}$$

Thus, any number system may be converted to our decimal system by using each number position as the multiplier for the base number raised to the power corresponding to that position.

Programmers for computers often use the octal base system for the simple reason that the conversion from binary to octal can be carried out mentally, and only a third as many digits are required to carry the same information. All that is necessary to convert from binary to octal is to group the binary bits in groups of three's, and write down the value of each group taken as an integer. For instance $(11111010110)_2$ would be written as $(011,111,010,110)_2 = (3726)_8$. Here $011 = 3$, $111 = 7$, $010 = 2$, and $110 = 6$, using the binary conversion system discussed above.

The simplicity of conversion could be accomplished using any base which is a power of 2. For instance, a base 4 could have been used, however, it would not eliminate the cumbersomeness of the large binary numbers. A base such as 16 or 32 could also have been used, but here it would have been necessary to invent new characters for digits over 9. So the octal base 8 system is often used for input-output information in computers.

*Use of the Octal System for THINK-A-DOT
and how it can count from 0 to 128*

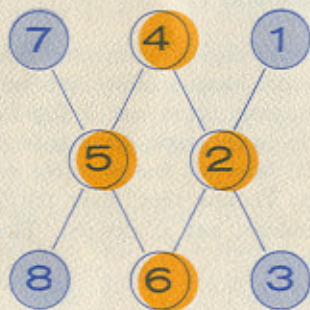
THINK-A-DOT is a binary digital computer that is most easily programmed with the octal base number system for input/output information. This is because there are 3 flip-flops in a vertical line. If we drop a marble in the right hand hole 8 times, the three right hand flip-flops will form all the possible color combinations without any duplication. To see this, first tilt THINK-A-DOT to the left and then, of course, set it upright. Then looking at the three right hand most colors -

From Top to Bottom they are	Blue	Yellow	Blue
Now, drop a marble in R and they become	Yellow	Yellow	Yellow
The second drop gives	Blue	Blue	Blue
The third drop gives	Yellow	Blue	Yellow
The fourth drop gives	Blue	Yellow	Yellow
The fifth drop gives	Yellow	Yellow	Blue

Blue	Blue	Yellow
Yellow	Blue	Blue
Blue	Yellow	Blue

0 - 101	5 - 001
1 - 000	6 - 110
2 - 111	7 - 011
3 - 010	8 - 101
4 - 100	

If we number the flip-flops as follows:



1 0 1 0 0 0 1 1

where the left hand bit corresponds to the #1 flip-flop and the

Since there are 8 flip-flops there should be 256 possible color combinations. However, according to Theorem 4, Page 7 of Section IV, only half of these patterns are attainable from the starting position. Thus, the next step will be to develop a table of binary code to represent the numbers from 0 to 128. The numbers from 0 to 7, will be as previously developed. To form the code for 8 however, an eighth marble will have to be dropped in R and an additional one will have to be dropped in M (the carry).

The table is as follows:

Decimal Number	Octal Number	Number of Marbles Dropped			Binary Code	123 456 78
		L	M	R		
0	0	0	0	0	101 000 11	
1	1	0	0	1	000 000 11	
2	2	0	0	2	111 000 11	
3	3	0	0	3	010 000 11	
4	4	0	0	4	100 001 11	
5	5	0	0	5	001 001 11	
6	6	0	0	6	110 001 11	
7	7	0	0	7	011 001 11	
8	10	0	1	0(8)*	110 100 11	
9	11	0	1	1	011 100 11	
10	12	0	1	2	101 101 11	
11	13	0	1	3	000 101 11	
12	14	0	1	4	111 101 11	
13	15	0	1	5	010 101 11	
14	16	0	1	6	100 100 11	
15	17	0	1	7	001 100 11	
16	20	0	2	0(8)*	110 011 11	
17	21	0	2	1	011 011 11	
18	22	0	2	2	101 010 11	
19	23	0	2	3	000 010 11	
20	24	0	2	4	111 010 11	
21	25	0	2	5	010 101 11	
22	26	0	2	6	100 011 11	
23	27	0	2	7	001 011 11	
24	30	0	3	0(8)*	100 110 11	
25	31	0	3	1	001 110 11	
26	32	0	3	2	110 110 11	
27	33	0	3	3	011 110 11	
28	34	0	3	4	101 111 11	
29	35	0	3	5	000 111 11	
30	36	0	3	6	111 111 11	
31	37	0	3	7	010 111 11	
32	40	0	4	0(8)*	100 000 10	
33	41	0	4	1	001 000 10	
34	42	0	4	2	110 000 10	
35	43	0	4	3	011 000 10	
36	44	0	4	4	101 001 10	
37	45	0	4	5	000 001 10	
38	46	0	4	6	111 001 10	
39	47	0	4	7	010 001 10	
40	50	0	5	0(8)*	111 100 10	
41	51	0	5	1	010 100 10	
42	52	0	5	2	100 101 10	
43	53	0	5	3	001 101 10	
44	54	0	5	4	110 101 10	
45	55	0	5	5	011 101 10	
46	56	0	5	6	101 100 10	
47	57	0	5	7	000 100 10	
48	60	0	6	0(8)*	111 011 10	
49	61	0	6	1	010 011 10	
50	62	0	6	2	100 010 10	
51	63	0	6	3	001 010 10	
52	64	0	6	4	110 010 10	
53	65	0	6	5	011 010 10	
54	66	0	6	6	101 011 10	
55	67	0	6	7	000 011 10	

Decimal Number	Octal Number	Number of Marbles Dropped			Binary Code
		L	M	R	
		1	2	3	4 5 6 7
56	70	0	7	0(8)*	101 110 10
57	71	0	7	1	000 110 10
58	72	0	7	2	111 110 10
59	73	0	7	3	010 110 10
60	74	0	7	4	100 111 10
61	75	0	7	5	001 111 10
62	76	0	7	6	110 111 10
63	77	0	7	7	011 111 10
64	100	1	0(8)	0(8)*	101 011 01
65	101	1	0	1	000 011 01
66	102	1	0	2	111 011 01
67	103	1	0	3	010 011 01
68	104	1	0	4	100 010 01
69	105	1	0	5	001 010 01
70	106	1	0	6	110 010 01
71	107	1	0	7	011 010 01
72	110	1	1	0(8)*	110 111 01
73	111	1	1	1	011 111 01
74	112	1	1	2	101 110 01
75	113	1	1	3	000 110 01
76	114	1	1	4	111 110 01
77	115	1	1	5	010 110 01
78	116	1	1	6	100 111 01
79	117	1	1	7	001 111 01
80	120	1	2	0(8)*	110 001 00
81	121	1	2	1	011 001 00
82	122	1	2	2	101 000 00
83	123	1	2	3	000 000 00
84	124	1	2	4	111 000 00
85	125	1	2	5	010 000 00
86	126	1	2	6	100 001 00
87	127	1	2	7	001 001 00
88	130	1	3	0(8)*	100 100 00
89	131	1	3	1	001 100 00
90	132	1	3	2	110 100 00
91	133	1	3	3	011 100 00
92	134	1	3	4	101 101 00

*The eighth marble drop is marked 0 again because the pattern for that hole repeats itself after every eight drops. This allows the number of marbles dropped in each hole to equal the corresponding octal number.

Try making some of the numbers listed in the Table. For instance, the decimal number $(113)_{10}$ is represented by the octal number $(161)_8$ which is formed (after first tilting to the left) simply by dropping 1 marble in L, 6 in M, and 1 in R. Keep practicing with the numbers until you understand how to form them. Note that in doing everything above you have been making THINK-A-DOT COUNT.

To make the transformations directly, notice, for instance, that $(128)_{10}$ is actually $(2) \times 64 = 2 \times 8^2$. The 8^2 is represented by the L hole. So drop (2) marbles in L and the number $(128)_{10}$ is formed. As another example, the number $(119)_{10}$ can be written $1 \times 64 + 6 \times 8 + 7 \times 1 = (1) \times 8^2 + (6) \times 8^1 + (7) \times 8^0$, so to make this number drop 1 in L, 6 in M, and 7 in R. Generally, the technique is to subtract out the largest power of 8 as many times as possible, then go to the next lower power of 8 and so on. This laborious process is not really necessary since the above conversion table may be used.

Addition with THINK-A-DOT

To add two decimal numbers first look up their octal equivalents in the table and then drop the corresponding summation of marbles in each hole. If the summation in a given hole requires 8 or more drops, subtract 8 from the number and drop (carry) 1 in the next higher order hole (always to the left).

As an example to add $(45)_{10} + (63)_{10}$

$$\begin{array}{r}
 (45)_{10} \\
 + (63)_{10} \\
 \hline
 (108)_{10}
 \end{array}
 \qquad
 \begin{array}{r}
 \text{Carry} \quad \begin{array}{cc} L & M \\ 1 & 1 \end{array} \\
 \begin{array}{cc} & (55)_8 \\ + & (77)_8 \end{array} \\
 \hline
 (154)_8
 \end{array}$$



To do the example with THINK-A-DOT, enter 55, by dropping 5 in R and 5 in M. Next drop 7 in R and 1 in M since there have been more than 8 marbles dropped in R. Lastly, drop 7 in M and 1 in L for the carry (since more than 8 marbles have also been dropped in M). Now look in the chart under $(108)_{10}$ and you will see the code for the pattern on THINK-A-DOT.

The above example had two carries. The following example has but one carry.

$$\begin{array}{r} (15)_{10} \\ + (36)_{10} \\ \hline (51)_{10} \end{array}$$

$$\begin{array}{r} \text{Carry } 1 \\ \text{M} \\ (17)_8 \\ + (44)_8 \\ \hline (63)_8 \end{array}$$



This would be done on THINK-A-DOT by entering $(17)_8$ as 7 marbles in R (after tilting to the *left*) and 1 marble in M. Then drop 4 marbles in R and 4 in M, plus one more in M for the carry of 1. Now look up the pattern (00101010) which you should have on the machine in the above table. The answer should, of course, be an octal $(63)_8$ or the decimal 51.

Try adding some more numbers. *Don't forget to carry 1* everytime more than 8 marbles have been dropped in *any* hole.

Subtraction

Subtraction in a computer is carried out by addition of complements. To understand the basis for this operation let us look at a problem of subtraction of decimal numbers.

$$493,201 - 126,944 = 493,201 + (1,000,000 - 126,944) - 1,000,000.$$

The subtraction $1,000,000 - 126,944$ can be very simply performed by subtracting each digit from 9, except the low order one, (in this case the right hand 4) which is subtracted from 10. The answer to this subtraction (873,056) is then called the *ten's complement*.

$$\text{Thus } 1,000,000 - 126,944 = 873,056.$$

$$\begin{array}{r} \text{Now adding } 493,201 \\ + 873,056 \\ \hline \end{array}$$



gives you 1,366,257.

The 1,000,000 is subtracted off simply by deleting the 1 at the beginning, giving the answer 366,257.

An alternate way of thinking of the same process involves what is known as the *nine's complement*. It is formed by subtracting each digit from 9 *including* the low order. After the addition of the complement, the leading 1 (from the 1,000,000 in our example) is deleted and a 1 is added to the units position (because we subtracted the low order digit from 9 instead of 10). This process is sometimes called an "end-around-carry."

In either method, if a larger number is subtracted from a smaller number, a negative difference will result. The negative difference will be in complement form and must be reconverted. In hand computations it is wise to be sure to complement the smaller of the two numbers and avoid this problem.

This may seem to be an exceedingly difficult way of subtracting two numbers. It is used (or the equivalent process in binary) because it is far simpler to form a digit-by-digit complement than to build circuits or devices to "borrow", as is ordinarily done in everyday decimal arithmetic. This is especially true of binary, where the one's complement is formed simply by changing ones to zeros and zeros to ones, which can be done with great ease electronically.

If the reader would like to read more thoroughly about computer manipulations of binary numbers, may we suggest DIGI-COMP I with its Advanced Instruction Manual. (See end of this Manual.)

Octal subtraction can be carried out in a similar manner. The first part of the operation is to form the "sevens complement." Then add in the usual fashion and perform the "end-around-carry." For instance, to subtract $(47603)_8$ from $(56162)_8$ first find

the sevens complement of the smaller number $(47603)_8$ which is $(30174)_8$ and add as follows:

$$\begin{array}{r} (56162)_8 \\ + (30174)_8 \text{ [complement of } (47603)_8 \text{]} \\ \hline (106356)_8 \end{array}$$

Next perform the "end-around-carry" by removing the high order 1 and adding it to the low order position. This will give the answer of $(6357)_8$. To check the answer simply add the answer to $(47603)_8$ as follows:



$$\begin{array}{r} (47603)_8 \\ + (6357)_8 \\ \hline (56162)_8 \end{array}$$

Try a few more problems to be sure you know how to do it. Now we are prepared to subtract using THINK-A-DOT. To illustrate the process, suppose we are to subtract $(67)_{10}$ from $(93)_{10}$ giving an answer of $(26)_{10}$. First look up the octal equivalents of 67 and 93 which are $(103)_8$ and $(135)_8$ respectively. Then find the sevens complement of the smaller number $(103)_8$ which is $(674)_8$. Now enter $(135)_8$ in THINK-A-DOT by dropping 1 in L, 3 in M, and 5 in R. Add to this $(674)_8$ by dropping 6 in L, 7 in M, and 4 in R. Remember that when you drop more than eight in a given hole, you must "carry" or drop another marble in the next higher order hole. Thus, the next thing to do is to drop another marble in each L and M. The last step is to "end-around-carry." So just drop one marble in R. (We didn't need to remove the high order 1, as it would have been in the fourth position which is non-existent.) To check your answer look up $(26)_{10}$ in the table and you will see the binary code for the pattern in your THINK-A-DOT.

As another example, suppose we subtract $(6)_{10}$ from $(14)_{10}$.

$$\begin{array}{r} 14 \\ -6 \\ \hline (8)_{10} \end{array} \quad \begin{array}{r} 16 \\ -6 \\ \hline (10)_8 \end{array} \quad \begin{array}{r} 16 \\ +1 \\ \hline 17 \\ +1 \\ \hline 18 \end{array} \begin{array}{l} \text{(sevens complement of 6)} \\ \\ \text{(end-around-carry)} \\ \text{(answer)} \end{array}$$

To do this problem enter $(16)_8$, add $(1)_8$, and "end-around-carry." This may be done simply by adding 1 in R. (In this case it is not wise to carry to M because it would just have to be subtracted out.) The best rule for handling the high order 1 is to just neglect to carry the highest order bit. A little practice will make this procedure clear.

On Page 15 we said that THINK-A-DOT could count to 128; however, the reader may form the numbers from 129 to 256 by first tilting THINK-A-DOT to the right and then, using a pencil, to hold the upper right hand flip-flop and tilting THINK-A-DOT to the left so that you get 001 000 11. You should now be able to add and subtract with numbers between $(129)_{10}$ and $(256)_{10}$.

ANSWERS

R-Right, L-Left, M-Middle

Answers to the THINK-A-DOT problems on Page 0. (Don't look at these unless you really have to.)

Answers on Page 0:
A (RRL), B (RL), C (RRL), D (RL), E (RRL), F (RRL), G (RRL), H (RRL), I (RRL), J (RRL), K (RRL), L (RRL), M (RRL), N (RRL), O (RRL), P (RRL), Q (RRL), R (RRL), S (RRL), T (RRL), U (RRL), V (RRL), W (RRL), X (RRL), Y (RRL), Z (RRL), 0 (RRL), 1 (RRL), 2 (RRL), 3 (RRL), 4 (RRL), 5 (RRL), 6 (RRL), 7 (RRL), 8 (RRL), 9 (RRL), A (RRL), B (RRL), C (RRL), D (RRL), E (RRL), F (RRL), G (RRL), H (RRL), I (RRL), J (RRL), K (RRL), L (RRL), M (RRL), N (RRL), O (RRL), P (RRL), Q (RRL), R (RRL), S (RRL), T (RRL), U (RRL), V (RRL), W (RRL), X (RRL), Y (RRL), Z (RRL), 0 (RRL), 1 (RRL), 2 (RRL), 3 (RRL), 4 (RRL), 5 (RRL), 6 (RRL), 7 (RRL), 8 (RRL), 9 (RRL).

OTHER COMPUTERS BY **ESR**

DIGI-COMP[®] 1

World's first DIGITAL COMPUTER entirely in plastic. Open so that you can see computer actions usually hidden in electronic circuits. DIGI COMP is entirely mechanical.

The remarkable 28-page Instruction Manual includes a simple explanation of binary arithmetic and computer fundamentals of operation and programming. Assembling DIGI COMP permits first hand observation of its components.



Watch DIGI COMP

Add Multiply	Countdown Answer Riddles	Program Set	Sequence Shift
-----------------	-----------------------------	----------------	-------------------

A total of 15 experiments are described.

DIGI COMP operates entirely in the binary mode, right before your eyes. That is why it is so simple to understand.

DIGI COMP teaches you to think and reason logically . . . A perfect math or science project . . . and have fun all the time.

Acclaimed by Educators and Scientists and Engineers . . . Tens of thousands have already been delighted by the now-famous DIGI COMP.

An advanced instruction manual is now available that presents the programming of each experiment in Computer Math (Boolean Algebra) and Flow Diagrams.

Fully guaranteed to please and educate\$5.98

FOR AGES 10 to ADULT (and Ph.D.)

The Amazing DR. NIM[™]

WHO is DR. NIM? He is a fascinating, challenging plastic digital computer. Ingeniously designed and simply made so that all can play against him. Less than a minute is needed to learn the rules of play.

If you can count, you can play DR. NIM. Or, if you wish, you may try to work out the mathematical formulas for beating him at the ancient game of NIM. This game, played for thousands of years, really comes to life with DR. NIM.



You will be completely fascinated when Dr. Nim himself AUTOMATICALLY plays several marbles as computed by his computer!

The instruction booklet that comes with DR. NIM not only describes the extremely simple rules of play, BUT it also presents a detailed write-up of DR. NIM as a COMPUTER. INCLUDED are discussions of the mathematics and the programming of computers. An intriguing discussion of the philosophy of computers as "Thinking Machines" is included.

DR. NIM was invented by a computer genius. That is why he is so simple to operate and yet so intriguing to play by both young and old. Match your wits against DR. NIM . . . He has been fully tested with children and adults. Guaranteed to please..... \$3.50

AGES 6 to ADULT (and Ph.D.)